

A Minimal Memory-Field AI Simulator with Self-Archiving Stability

Daniel Jacob Read IV
Independent Research

December 2025

Abstract

This document presents a minimal computational prototype inspired by Inward Physics, modeling artificial intelligence memory as a continuously evolving scalar field. The system explores how diffusion, relaxation, and attention-driven amplification interact, and demonstrates an instability arising from unbounded amplification. A simple saturation mechanism stabilizes the dynamics, enabling coherent long-term behavior and a basic form of self-archiving. The prototype is exploratory and does not claim artificial consciousness, but illustrates intrinsic stability mechanisms relevant to long-running AI systems.

1 Motivation

Modern artificial intelligence systems rely on static checkpoints, external logging, and re-training to manage drift. These approaches treat memory as passive rather than dynamic. Long-running or adaptive systems face a deeper challenge: preserving internal coherence while continuing to evolve.

This work explores a toy alternative by modeling memory as a continuous field governed by simple physical-style constraints. The goal is not biological realism, but to test whether intrinsic stability and self-archiving behavior can emerge from minimal structure.

2 Memory-Field Model

Memory is represented as a scalar field

$$\mu(x, t) \in \mathbb{R}^+$$

defined over a one-dimensional spatial domain.

The evolution equation is:

$$\frac{\partial \mu}{\partial t} = D \nabla^2 \mu - \lambda(\mu - \mu_0) + S(x, t)$$

where:

- D controls memory diffusion,
- λ enforces relaxation toward a baseline μ_0 ,
- $S(x, t)$ represents attention-driven amplification.

3 Prototype Implementation

The system is initialized with a localized “memory well” represented by a Gaussian peak above baseline. Attention is modeled as intermittent, spatially focused source pulses. A self-archiving mechanism records field states when variance drops below a coherence threshold.

4 Observed Instability

Initial simulations exhibited catastrophic numerical instability. Memory density grew without bound, integrated mass overflowed floating-point limits, and coherence metrics became meaningless. This failure mode reflects unbounded positive feedback in the attention term and mirrors exploding-gradient problems in neural networks.

5 Stabilization via Saturation

To prevent runaway growth, a saturation term was introduced:

$$S(x, t) \rightarrow S(x, t) \left(1 - \frac{\mu}{\mu_{\max}} \right)$$

This preserves amplification at low density while enforcing a natural upper bound. Combined with stable time stepping and non-negativity constraints, the system evolves smoothly and predictably.

6 Results

With saturation enabled, memory density remains finite, mass stabilizes, and variance evolves smoothly. Self-archiving triggers only when genuine coherence is achieved rather than as an artifact of numerical failure.

7 Limitations

This model is intentionally minimal. It operates in one dimension, uses explicit integration, and contains no learning or adaptive parameters. No claims are made regarding consciousness or subjective experience.

8 Future Directions

Extensions may include higher-dimensional fields, differentiable implementations, entropy-based coherence metrics, and integration with AI system diagnostics.

9 Conclusion

This prototype demonstrates that memory-field-inspired systems can be implemented, can fail in predictable ways, and can be stabilized using standard techniques. The value lies in translating abstract ideas into operational constraints rather than making expansive claims.

Appendix A: Reference Python Implementation

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 # Parameters
5 D = 0.05
6 lam = 0.15
7 mu0 = 0.10
8 mu_max = 2.0
9
10 # Grid
11 N = 400
12 x = np.linspace(-10, 10, N)
13 dx = x[1] - x[0]
14
15 # Initial condition
16 mu = mu0 + 1.2 * np.exp(-x**2 / (2 * 0.9**2))
17
18 def laplacian(u):
19     return (np.roll(u,1) - 2*u + np.roll(u,-1)) / dx**2
20
21 dt = 0.45 * dx*dx / (2*D)
22
23 for t in range(400):
24     A = 0.25 * np.exp(-x**2 / (2*0.8**2))
25     S = A * (1 - mu / mu_max)
26     dmu = D*laplacian(mu) - lam*(mu - mu0) + S
27     mu += dt * dmu
28     mu = np.clip(mu, 0, mu_max)
29
30 plt.plot(x, mu)
31 plt.title("Memory Field (x) at Final Time")
32 plt.xlabel("x")
33 plt.ylabel("(x)")
34 plt.show()
```

A Minimal Memory-Field AI Simulator with Self-Archiving Stability

Daniel Jacob Read IV
Independent Research
December 2025

© 2025 Daniel Jacob Read IV.
All rights reserved.

No part of this document may be reproduced, distributed, or transmitted in any form or by any means, including electronic or mechanical methods, without prior written permission of the author, except for brief quotations used in scholarly review or citation.

This work is provided for research and exploratory purposes only. No claims are made regarding artificial consciousness or subjective experience.